June 1988 DTIC FILE COPY UILU-ENG-88-2227
ACT-92

(2)

AD-A197 051

COORDINATED SCIENCE LABORATORY
College of Engineering
Applied Computation Theory

DYNAMIC MAINTENANCE OF PLANAR DIGRAPHS, WITH APPLICATIONS

Roberto Tamassia
Franco Preparata

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-88-2227 (ACT-92)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research; National Science Foundation
6c. ADDRESS (City, State, and ZIP Code) 1101 W. Springfield Ave. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217 1800 G. Street, N.W. Washington, D.C. 20550
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Joint Services Electronics Program & NSF 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-84-C-0149 ECS-84-10902
8c. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217 1800 G. Street, N.W. Washington, D.C. 20550	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Dynamic Maintenance of Planar Digraphs, with Applications

12. PERSONAL AUTHOR(S) Tamassia, Roberto
Preparata, Franco

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM ____ TO ____	1988 June	27

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	planar, *st*-graph transitive closure, point location, contact-chain, planar subdivision, dynamic data structure, on-line algorithm

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

We show the a planar *st*-graph G admits two total orders (called leftist and rightist, respectively) on the set $V \cup E \cup F$, where V, E, and F are respectively the set of vertices, edges, and faces of G, with $|V| = n$. Assuming that G is to be dynamically modified by means of insertions of edges and expansions of vertices (and their inverses), we exhibit a $O(n)$-space dynamic data structure for the maintenance of these orders such that an update can be performed in time $O(\log n)$. The discovered structural properties of planar *st*-graphs provide a unifying theoretical underpinning for several applications, such as dynamic point location in planar monotone subdivisions, dynamic transitive-closure query in planar *st*-graphs, and dynamic contact-chain query in convex subdivisions. The presented techniques significantly outperform previously known solutions of the same problems.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

Dynamic Maintenance of Planar Digraphs, with Applications *

Roberto Tamassia and Franco P. Preparata

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1101 W. Springfield
Urbana, IL 61801

Abstract

a certain set

We show the a planar *st*-graph G admits two total orders (called leftist and rightist, respectively) on ~~the~~ set $V \cup E \cup F$, where V, E, and F are respectively the set of vertices, edges, and faces of G, with $|V| = n$. Assuming that G is to be dynamically modified by means of insertions of edges and expansions of vertices (and their inverses), we exhibit a $O(n)$-space dynamic data structure for the maintenance of these orders such that an update can be performed in time $O(\log n)$. The discovered structural properties of planar *st*-graphs provide a unifying theoretical underpinning for several applications, such as dynamic point location in planar monotone subdivisions, dynamic transitive-closure query in planar *st*-graphs, and dynamic contact-chain query in convex subdivisions. The presented techniques significantly outperform previously known solutions of the same problems.

Key words: planar *st*-graph, transitive closure, point location, contact-chain, planar subdivision, dynamic data structure, on-line algorithm.

* This work was supported in part by National Science Foundation Grant ECS-84-10902 and by the Joint Services Electronics Program under Contract N00014-84-C-0149.

1. Introduction

The notion of a planar *st*-graph – i.e., a planar acyclic digraph embedded in the plane with exactly one source, s, and one sink, t, both on the external face – was first introduced in the planarity testing algorithm of Lempel *et al.*[18], and was fruitfully used in a number of applications, which include planar graph embedding [4, 13, 27], graph planarization [12, 20], graph drawing algorithms [5, 26, 31], floor planning [1, 29] planar point location [6, 17], visibility representations [19, 24, 25, 30], motion planning [8, 23], and VLSI layout compaction [9, 29]. Also, planar *st*-graphs are important in the theory of partially ordered sets since they are associated with planar lattices [15].

In this paper we further the investigation of these structures, and show that any planar *st*-graph G admits two total orders (referred to as leftist and rightist orders) on the set $V \cup E \cup F$, where V, E, and F are respectively the set of vertices, edges, and faces of G. Each of these two orders yields a unique representation of G as a string of all its topological constituents. Graph G can be dynamically modified by means of insertion of edges and expansions of vertices, and of their inverses. These operations form a complete set, since any n-vertex planar *st*-graph can be assembled or disassembled by an appropriate sequence of $O(n)$ such operations.

The central result of this paper is that the string representation of the graph resulting from one of the postulated updating operations is obtained as a syntactic transformation of the pre-update string representation. This transformation consists of the execution of $O(1)$ primitives, such as insertions, deletions, and swaps of substrings.

This general framework provides the theoretical underpinning and unifying viewpoint for three significant applications: point location in a planar monotone subdivision, transitive-closure query in planar *st*-graphs, and contact-chain query in convex subdivisions. In this paper we shall only briefly illustrate (in Section 4) the connection between planar *st*-graphs and monotone subdivisions, since the point location problem in the latter has been treated earlier in exclusively geometric terms and is reported elsewhere [22]. We simply recall that a *monotone subdivision* Γ is a partition of the plane into regions that are *monotone polygons*, (i.e., polygons whose intersection with a fixed direction – e.g., horizontal – consists of at most one segment). The point location problem in Γ consists of finding the region containing a query point q. The main result of [22], reported here for completeness, is expressed by the following theorem:

Theorem A: Let Γ be a monotone planar subdivision with n vertices. There exists an $O(n)$-space dynamic point location data structure with query time $O(\log^2 n)$, which allows for insertion/deletion of a vertex in time $O(\log n)$ and insertion/deletion of a chain of k edges in time $O(\log^2 n + k)$ (worst-case).

A *transitive-closure query* for a planar *st*-graph G consists of testing for the existence of (and/or reporting) a directed path between two vertices u and v of G. We are interested in a graph G that can be dynamically modified.

The previous best results concern semi-dynamic versions of this problem (where only either insertions or deletions of edges are allowed), and have $O(1)$ query time, $O(n)$ *amortized* update time, and $O(n^2)$ storage [10, 11]. In this paper we establish the following result:

Theorem B: Let G be a planar *st*-graph with n vertices. There exists an $O(n)$-space dynamic data structure for the transitive-closure query problem on G, which supports queries and updates in time $O(\log n)$ (worst-case).

Finally, we consider the problem of contact-chain query in convex subdivisions, which arises in motion planning and computer graphics, and is described as follows [3, 8, 23]. Given a *convex subdivision* Γ of the plane (note that a convex subdivision is a special case of monotone subdivision) and an (oriented) direction θ, we say that region r_1 *pushes* an adjacent region r_2 if there exists a line in direction θ which intersects r_1 and r_2 in that order. A *contact chain* in Γ is a sequence of regions $r_1, r_2, \cdots, r_k$ such that r_i pushes r_{i+1} for $i = 1, \cdots, k-1$ (see Fig. 1). Assume that the regions of Γ are rigid objects, and we want to translate them one at a time in direction θ avoiding collisions. Then the existence of a contact chain from r_1 to r_2 implies that r_2 obstructs r_1, i.e., r_2 must be translated before r_1.

A *contact-chain query* consists of testing the existence of (and/or reporting) a contact chain between two regions of Γ. We are interested in answering contact-chain queries in a very general dynamic environment where Γ can be updated by means of insertion/deletions of vertices and edges, and the direction θ can be changed by *elementary* increments/decrements. (An elementary increment/decrement of direction is such that the push relation is inverted in exactly one pair of adjacent regions.) Casting this problem in the planar *st*-graph framework, we establish

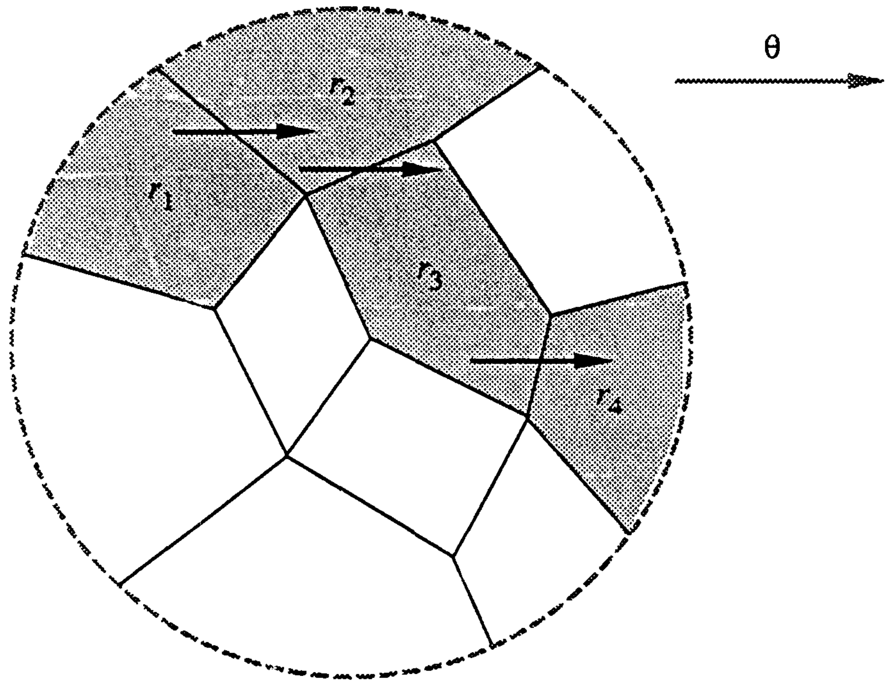

Figure 1 Example of contact chain.

the following result:

Theorem C: Let Γ be a convex subdivision with n vertices. There exists an $O(n)$-space dynamic data structure for the contact-chain query problem in Γ, which supports queries and updates in time $O(\log n)$ (worst-case).

The rest of this paper is organized as follows. Section 2 provides preliminary definitions and properties of planar *st*-graphs. In Section 3 we present the technique for the dynamic maintenance of planar *st*-graphs. Applications to planar point location, transitive closure, and contact chains are described in Section 4.

2. Planar *st*-graphs

Basic definitions on graphs and posets can be found in textbooks such as [2, 7].

Let G be a directed graph, for brevity digraph, and v a vertex of G. We denote by $\deg^-(v)$ the *indegree* of v, i.e. the number of incoming edges of v, and by $\deg^+(v)$ the *outdegree* of v, i.e. the number of outgoing edges of v. A *source* of G is vertex s with $\deg^-(s)=0$. A *sink* of G is vertex t with $\deg^+(t)=0$. A *transitive* edge of G is an edge $e=(u,v)$ such that there exists another directed path from u to v consisting of at least two edges.

Definition 1 A *planar st-graph* is a planar acyclic digraph G with exactly one source, s, and exactly one sink, t, which is embedded in the plane so that s and t are on the boundary of the external face (see Fig. 2).

These graphs were first introduced in the planarity testing algorithm of Lempel *et al.*[18]. Several important properties of planar *st*-graphs are expressed by the following lemmas:

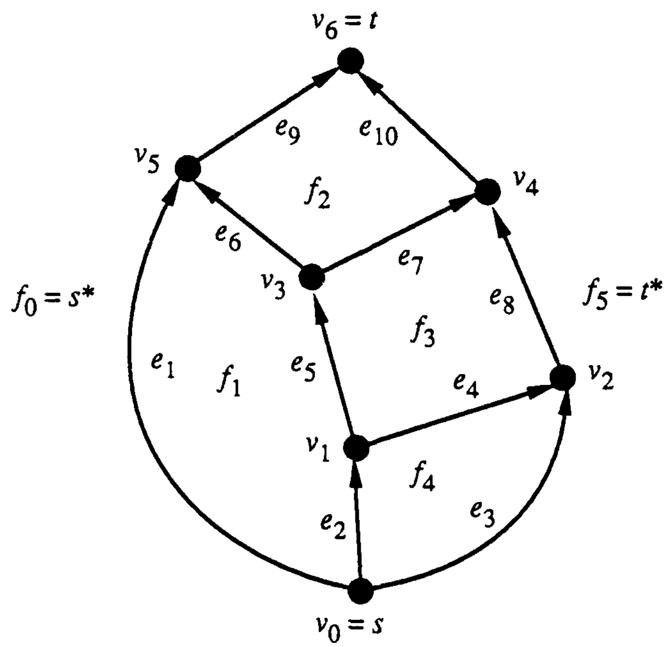

Figure 2 Example of planar *st*-graph.

Lemma 1 [18] Every vertex of G is on some directed path from s to t.

Lemma 2 [25] For every vertex v of G, the incoming (outgoing) edges appear consecutively around v (See Fig. 3.a).

Lemma 3 [25] For every face f of G, the boundary of f consists of two directed paths with common origin and destination. (See Fig. 3.b).

Lemma 4 [5, 16] G admits a planar *upward* drawing, i.e. a planar drawing such that every edge (u,v) is a curve monotonically increasing in the vertical direction.

Let P be a poset (partially ordered set), where $\ll$ denotes the partial order on the elements of P. The *Hasse diagram* (also called *covering digraph*) of P is a digraph G whose vertices are the elements of P, and such that (u,v) is an edge of G if and only if $u \ll v$ and there is no other element x of P such that $u \ll x \ll v$. G is acyclic and has no transitive edges. Hasse diagrams are

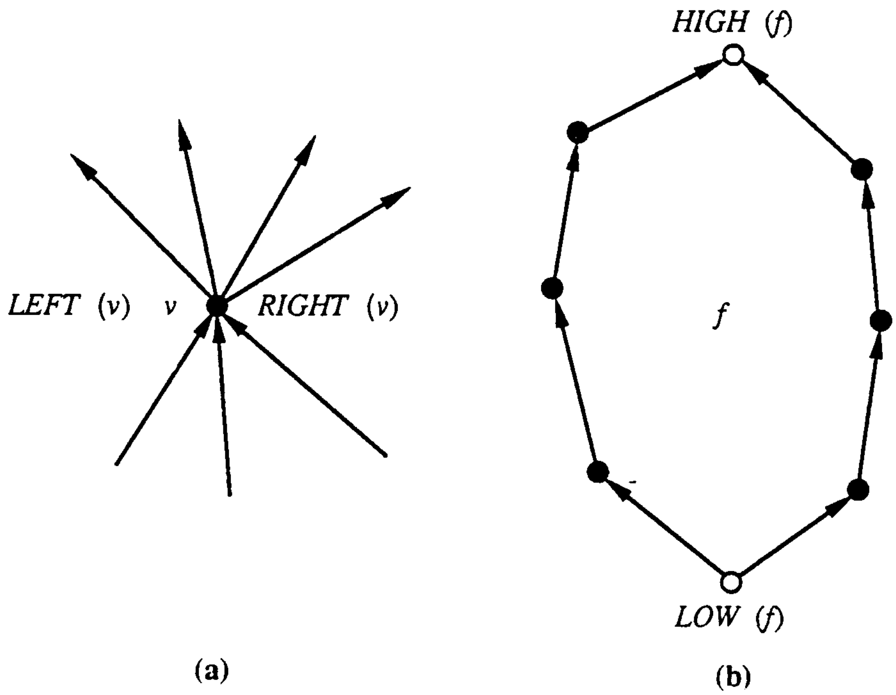

Figure 3 (a) Example for Lemma 2; (b) Example for Lemma 3.

usually represented by straight-line drawings such that for each edge (u,v) the ordinate of vertex u is smaller that that of vertex v.

A *planar lattice* is a poset whose Hasse diagram is a planar *st*-graph. Also, every plane *st*-graph without transitive arcs is the Hasse diagram of some planar lattice. Several properties of planar lattices are described in [15].

A *linear extension* of a poset P is a total order $<$ on the elements of P such that for any two elements u and v of P $u \ll v$ implies $u < v$. A linear extension corresponds to a topological sorting of the vertices of the Hasse diagram of P. We say that P has *dimension* k if G admits k linear extensions $<_1, <_2, \cdots, <_k$, such that $u \ll v$ if and only if $u <_1 v$, $u <_2 v$, $\cdots$, $u <_k v$, and k is minimum.

It is known that planar lattices have dimension 2 [2, p. 32, ex. 7(c)] [14, 15], which implies the following lemma:

Lemma 5 [2, 14, 15] Let G be a planar *st*-graph with n vertices. There exist two total orders on the vertices of G, denoted $<_L$ and $<_R$, such that there is a directed path from u to v if and only if $u <_L v$ and $u <_R v$. Furthermore, orders $<_L$ and $<_R$ can be computed in $O(n)$ time.

Lemma 5 is based on the fact that the underlying partial order of a planar lattice admits a ''complementary'' partial order (see [15]). Figure 4.a shows a planar *st*-graph where each vertex is labeled by its ranks in the orders $<_L$ and $<_R$.

In the following definitions, the concepts of *left* and *right* refer to the orientation of the edges. For example, the face to the left of an edge (u,v) is the face containing edge e which appears on the left side when traversing edge (u,v) from vertex u to vertex v. Also, the reader will find it convenient to visualize the planar *st*-graph G as being drawn in the plane with edges monotonically increasing in the vertical direction (see Lemma 4).

Given vertices u and v of G such that there exists a path from u to v, the set of paths from u to v defines a planar *st*-graph with source u and sink v which is an induced subgraph of G. The two paths that form the external boundary of this subgraph will be called the *leftmost path* and *rightmost path* from u to v, respectively. For example, the external boundary of G consists of the leftmost and rightmost paths from s to t.

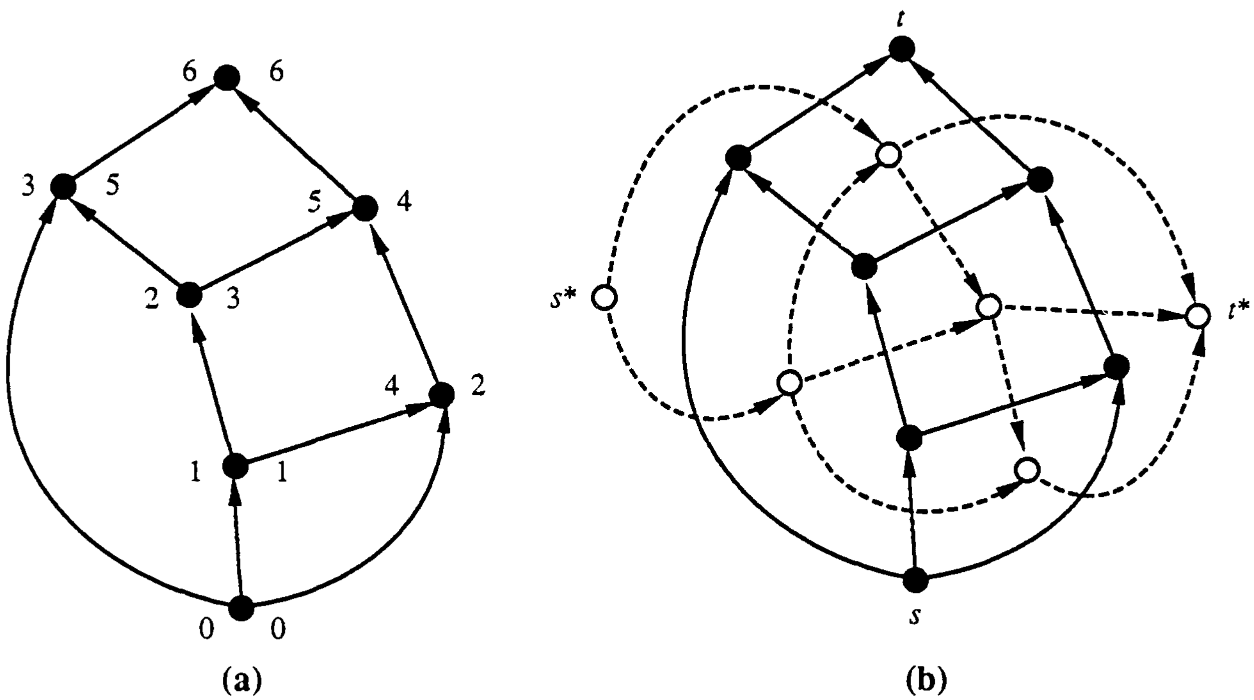

Figure 4 (a) Orders $<_L$ and $<_R$ on the vertices of a planar st-graph; (**b**) A planar st-graph G and its dual G^*.

Let G^* be the digraph obtained from the dual graph of G as follows (see Fig. 4.b): (1) the dual edge e^* of an edge e is directed from the face to the left of e to the face to the right of e; (2) the external face of G is dualized to two vertices of G^*, denoted s^* and t^*, which are incident with the duals of the edges on the leftmost and rightmost paths from s to t, respectively. Vertices s^* and t^* can be thought of as being the "left" and "right external face" of G, respectively. It is simple to verify that G^* is a planar st-graph with source s^* and sink t^* [19,25]. Notice that G^* might have multiple arcs.

Let V, E, and F denote the set of vertices, edges, and faces of G, respectively, where F has elements s^* and t^* representing the external face. We will show that the orders $<_L$ and $<_R$ can be extended to the set $V \cup E \cup F$, thereby giving a unique total order of all topological constituents of G.

First, for each element x of $V \cup E \cup F$, we define vertices $LOW(x)$ and $HIGH(x)$, and faces $LEFT(x)$ and $RIGHT(x)$, as follows:

(1) If $x = v \in V$, we define $LOW(v) = HIGH(v) = v$. Also, with reference to Lemma 2 and Fig. 3.a, we denote by $LEFT(v)$ and $RIGHT(v)$ the two faces that separate the incoming and outgoing edges of a vertex $v \neq s, t$. For $v = s$ or $v = t$, we conventionally define $LEFT(v) = s^*$ and $RIGHT(v) = t^*$.

(2) If $x = e \in E$, we define $LOW(e)$ and $HIGH(e)$ as the tail and head vertices of e, respectively. Also, we denote by $LEFT(e)$ and $RIGHT(e)$ the faces on the left and right side of e, respectively.

(3) If $x = f \in F$ and $f \neq s^*, t^*$, we denote by $LOW(f)$ and $HIGH(f)$ the two vertices that are the common origin and destination of the two paths forming the boundary of f (see Lemma 3 and Fig. 3.b). For $f = s^*$ or $f = t^*$, $LOW(f)$ and $HIGH(f)$ are undefined. Also, we define $LEFT(f) = RIGHT(f) = f$.

Definition 2 We say that x is *below* y, denoted $x \uparrow y$, if there is a path in G from $HIGH(x)$ to $LOW(y)$. Also, we say that x is *to the left of* y, denoted $x \rightarrow y$, if there is a path in G^* from $RIGHT(x)$ to $LEFT(y)$.

For example, in the planar *st*-graph shown in Fig. 2, we have $e_2 \uparrow v_4$, $f_4 \uparrow v_4$, $v_5 \rightarrow f_4$, and $e_1 \rightarrow f_2$.

Lemma 6 Relations $\uparrow$ and $\rightarrow$ are partial orders on $V \cup E \cup F$.

Proof: A consequence of the fact the graphs G and G^* are acyclic. $\square$

The following lemma shows that $\uparrow$ and $\rightarrow$ are complementary partial orders.

Lemma 7 Let x and y be any two elements of $V \cup E \cup F$. Then one and only one of the following holds:

$$x \uparrow y, \quad y \uparrow x, \quad x \rightarrow y, \quad y \rightarrow x.$$

Proof: We prove the theorem for the case when $y = v$ is a vertex of G. The other two cases can be proved using similar arguments.

Let π_1 and π_2 be the leftmost and rightmost paths from s to v, respectively. Also, let π_3 and π_4 be the leftmost and rightmost paths from v to t, respectively. These paths partition $V \cup E \cup F$ into five subsets, one of which is v, and the others are defined as follows (see Fig. 5):

(1) A contains the vertices, edges, and faces enclosed by paths π_1 and π_2, including the vertices and edges of these paths, but excluding v;

(2) B contains the vertices, edges, and faces enclosed by paths π_3 and π_4, including the vertices and edges of these paths, but excluding v;

(3) C contains the vertices, edges, and faces to the left of paths π_1 and π_3, excluding the vertices and edges of these paths;

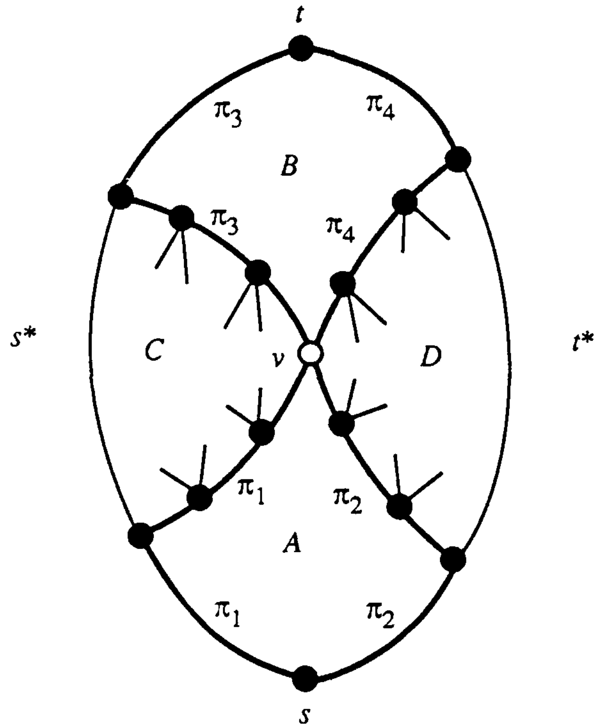

Figure 5 Partiton of $V \cup E \cup F$ with respect to vertex v.

(4) D contains the vertices, edges, and faces to the right of paths π_2 and π_4, excluding the vertices and edges of these paths.

It is easy to verify that the edges of A are those of a planar st-graph with source s and sink v, which is an induced subgraph of G, and, similarly, the edges of B are those of a planar st-graph with source v and sink t. Notice that the vertices w of A are exactly those such that there is a directed path in G from w to v, and analogously for the vertices of B.

Using simple duality arguments, we can show that the duals of the edges of C are those of a planar st-graph with source s^* and sink $LEFT(v)$, which is an induced subgraph of G^*. Similary, the duals of the edges of D are those of a planar st-graph with source $RIGHT(v)$ and sink t^*. Notice that the faces f of C are exactly those such that there is a directed path in G^* from f to $LEFT(v)$, and analogously for the faces of D.

By Lemma 1, there are directed paths from every vertex of A to v, and from v to every vertex of B. Since for every edge or face x of A (B), both $LOW(x)$ and $HIGH(x)$ are in A (B), we conclude that $x \in A$ implies $x \uparrow v$ and $x \in B$ implies $v \uparrow x$. With similar arguments, we conclude that $x \in C$ implies $x \rightarrow v$ and $x \in D$ implies $v \rightarrow x$.

It remains to be shown that relations $\uparrow$ and $\rightarrow$ are mutually exclusive. Let $x \in A \cup B$, i.e., either $x \uparrow v$ or $v \uparrow x$. Suppose $x \uparrow v$; if $x \rightarrow v$, then there is a path in G^* from $RIGHT(x)$ to $LEFT(v)$. This implies that $RIGHT(x) \in C$, a contradiction. An analogous contradiction is reached if we assume that $x \uparrow v$ and $v \rightarrow x$ jointly hold. Finally, let $x \in C \cup D$, i.e., either $x \rightarrow v$ or $v \rightarrow x$. Suppose $x \rightarrow v$; if $x \uparrow v$, then there is a path in G from $HIGH(x)$ to v. This implies that $HIGH(x) \in A$, a contradiction. An analogous contradiction is reached if we assume that $x \rightarrow v$ and $v \uparrow x$ jointly hold. □

Definition 3 We define relations $<_L$ and $<_R$ on $V \cup E \cup F$, as follows:

$$x <_L y \Leftrightarrow x \uparrow y \text{ or } x \rightarrow y; \quad x <_R y \Leftrightarrow x \uparrow y \text{ or } y \rightarrow x.$$

As a consequence of Lemma 7, we obtain:

Theorem 1 The relations $<_L$ and $<_R$ on $V \cup E \cup F$ are total orders.

We also note that there is a path in G from vertex u to vertex v if and only if $u <_L v$ and $u <_R v$, since such path exists if and only if $u \uparrow v$.

Definition 4 We define the *left-sequence* of G as the sequence of elements of $V \cup E \cup F$, sorted according to $<_L$ (leftist order). The *right-sequence* of G is defined similarly with respect to $<_R$ (rightist order).

For example, the right-sequence of the graph of Fig. 2 is:

$$f_5 v_0 e_3 f_4 e_2 v_1 e_4 v_2 e_8 f_3 e_5 v_3 e_7 v_4 e_{10} f_2 e_6 f_1 e_1 v_5 e_9 v_6 f_0.$$

We will use a convenient string notation for such sequences. Namely, we use terminal symbols (lower-case letters) for the elements of $V \cup E \cup F$, and variables (upper-case letters) for substrings of the left- or right-sequence. For example, the left-sequence of the graph of Fig. 2 can be represented by the string

$$f_0 v_0 e_1 A v_3 e_6 v_5 e_{10} f_2 B$$

where $A = f_1 e_2 v_1 e_5$ and $B = e_7 f_3 e_4 f_4 e_3 v_2 e_8 v_4 e_9 v_6 f_5$.

3. On-Line Maintenance of a Planar *st*-graph

In this section we define a complete set of update operations on a planar *st*-graph, and show that the restructuring of the orders $<_L$ and $<_R$ resulting from any such update operation can be expressed by means of a simple string transformation. From this result, we derive an efficient data structure for the on-line maintenance of the two orders of a planar *st*-graph.

The update operations on a planar *st*-graph are defined as follows:

INSERT $(e,u,v,f;f_1,f_2)$: Add edge $e=(u,v)$ inside face f, which is decomposed into faces f_1 and f_2, with f_1 to the left of e and f_2 to the right (see Fig. 6.a).

DELETE $(e,u,v,f_1,f_2;f)$: Delete edge $e=(u,v)$ and merge the two faces f_1 and f_2 formerly on the two sides of e into a new face f (see Fig. 6.a).

EXPAND $(e,f,g,v;v_1,v_2)$: Expand vertex v into vertices v_1 and v_2, which are connected by a new edge e with face f to its left and face g to its right (see Fig. 6.b).

CONTRACT $(e,f,g,v_1,v_2;v)$: Contract edge $e=(v_1,v_2)$, and merge its endpoints into a new vertex v. Faces f and g are to the left and right of e, respectively (see Fig. 6.b). Parallel edges resulting from the contraction are merged into a simple edge.

Each operation is allowed if the resulting graph is itself a planar *st*-graph. It is interesting to observe that operations *EXPAND* and *CONTRACT* are dual of *INSERT* and *DELETE*, respectively, since performing one on G corresponds to performing the other on G^*.

We say that an edge e of G is *removable*, if operation *DELETE* $(e,u,v,f_1,f_2;f)$ on G yields a planar *st*-graph. We say that e is *contractible* if operation *CONTRACT* $(e,f,g,v_1,v_2;v)$ on G yields a planar *st*-graph.

Lemma 8 Each edge of G is either removable or contractible.

Proof: From Definition 1, it is easy to see that an edge $e=(u,v)$ is removable if and only if $\deg^+(u) \geq 2$ and $\deg^-(v) \geq 2$, and it is contractible if and only if it is not a transitive edge. Assume that edge $e=(u,v)$ is not removable. Then we have $\deg^+(u)=1$ and/or $\deg^-(v)=1$. This implies that there is no other path in G from u to v, so that e cannot be a transitive edge. Hence, edge e is contractible. Conversely, assume that edge $e=(u,v)$ is not contractible. Then e

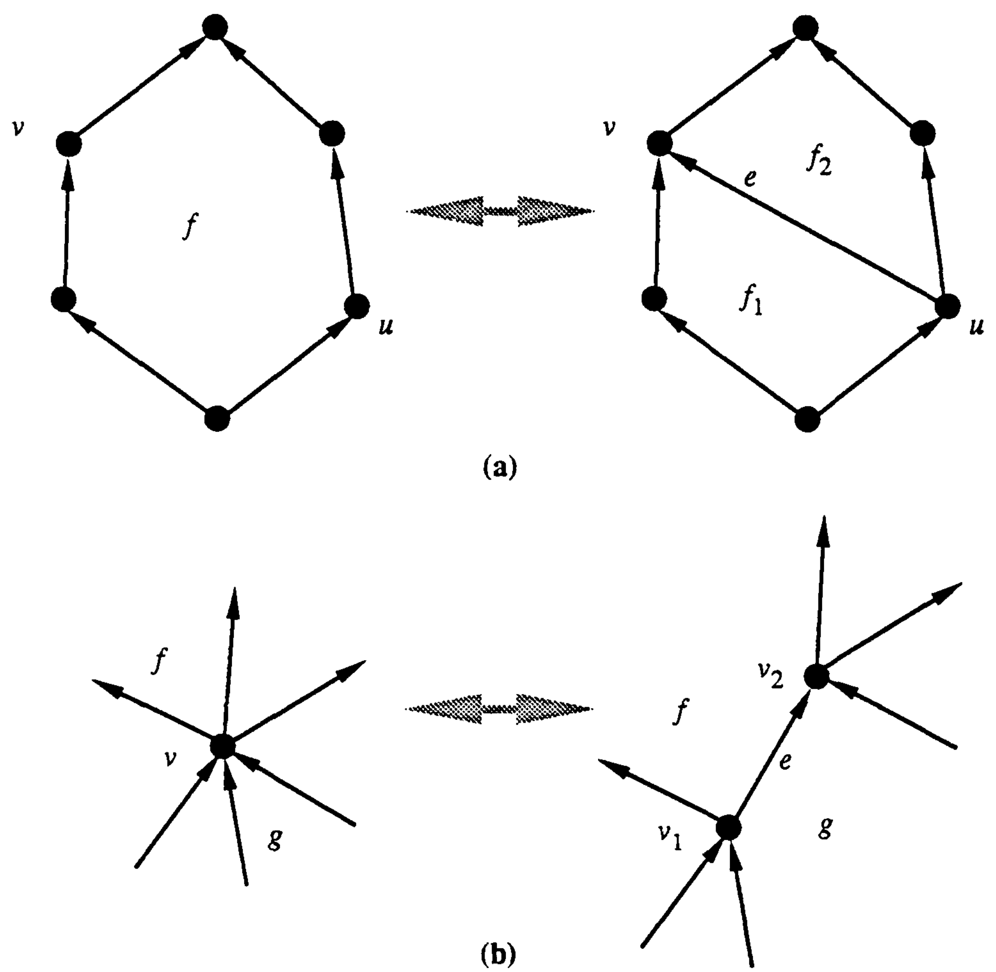

Figure 6 **(a)** Operations *INSERT* and *DELETE*; **(b)** Operations *EXPAND* and *CONTRACT*.

is a transitive edge, which implies $\deg^+(u) \geq 2$ and $\deg^-(v) \geq 2$, so that e is removable. $\square$

A simple induction based on Lemma 8 yields:

Lemma 9 Let G_0 be the trivial planar *st*-graph consisting of a single vertex. Any planar *st*-graph with n vertices can be assembled starting form G_0 by means of $O(n)$ *INSERT* and *EXPAND* operations, and can be disassembled to yield G_0 by means of $O(n)$ *DELETE* and *CONTRACT* operations.

Now, we describe the transformation of the leftist order $<_L$ as a consequence of operations *INSERT* $(e,u,v,f;f_1,f_2)$. Similar arguments hold for the order $<_R$ and for operation *EXPAND* $(e,f,g,v;v_1,v_2)$.

Theorem 2 Let G be a planar *st*-graph, and G' be the graph obtained from G after the execution of operation *INSERT* $(e,u,v,f;f_1,f_2)$. Depending on the relative orders of u, v, and f we have the following transformations (*left-sequence of* G) $\Rightarrow$ (*left-sequence of* G'):

(1) $u <_L v <_L f: \quad A\,u\,B\,v\,C\,f\,D \Rightarrow A\,u\,B\,f_1\,e\,v\,C\,f_2\,D;$

(2) $f <_L u <_L v: \quad A\,f\,B\,u\,C\,v\,D \Rightarrow A\,f_1\,B\,u\,e\,f_2\,C\,v\,D;$

(3) $u <_L f <_L v: \quad A\,u\,B\,f\,C\,v\,D \Rightarrow A\,u\,B\,f_1\,e\,f_2\,C\,v\,D;$

(4) $v <_L f <_L u: \quad A\,v\,B\,f\,C\,u\,D \Rightarrow A\,f_1\,C\,u\,e\,v\,B\,f_2\,D.$

Proof: The four cases are illustrated in Fig. 7. First, we observe that the union of the elements of $V \cup E \cup F$ associated with any one of the substrings A, B, C, and D, is a topologically connected region of the plane. The above regions, together with u, v, and f, form a partition of the entire plane, which is determined by the leftmost path from $HIGH(f)$ to t, the rightmost path from s to $LOW(f)$, and, depending respectively on each of the four cases, the following paths:

(1) the leftmost paths from u to t and from v to t (see Fig. 7.a,b);

(2) the rightmost paths from s to u and from s to v (see Fig. 7.c,d);

(3) the leftmost path from u to t and the rightmost path from s to v (see Fig. 7.e,f);

(4) the leftmost path from v to t and the rightmost path from s to u (see Fig. 7.g,h).

We discuss in detail Case 4 (see Fig. 7.g,h). The proof for the other cases can be derived with similar arguments. The insertion of edge e causes every vertex in C to be connected with a directed path to every vertex of B. At the same time, the insertion of e breaks all the paths of G^* from the faces of B to the faces of C. Hence, we have the following relations:

$$A <_L f_1,\ f_1 \rightarrow C,\ C \uparrow u,\ u \uparrow e,\ e \uparrow v,\ v \uparrow B,\ B \rightarrow f_2,\ f_2 <_L D,$$

where a substring represents compactly all of its elements. These relations yield immediately the updated left-sequence. $\square$

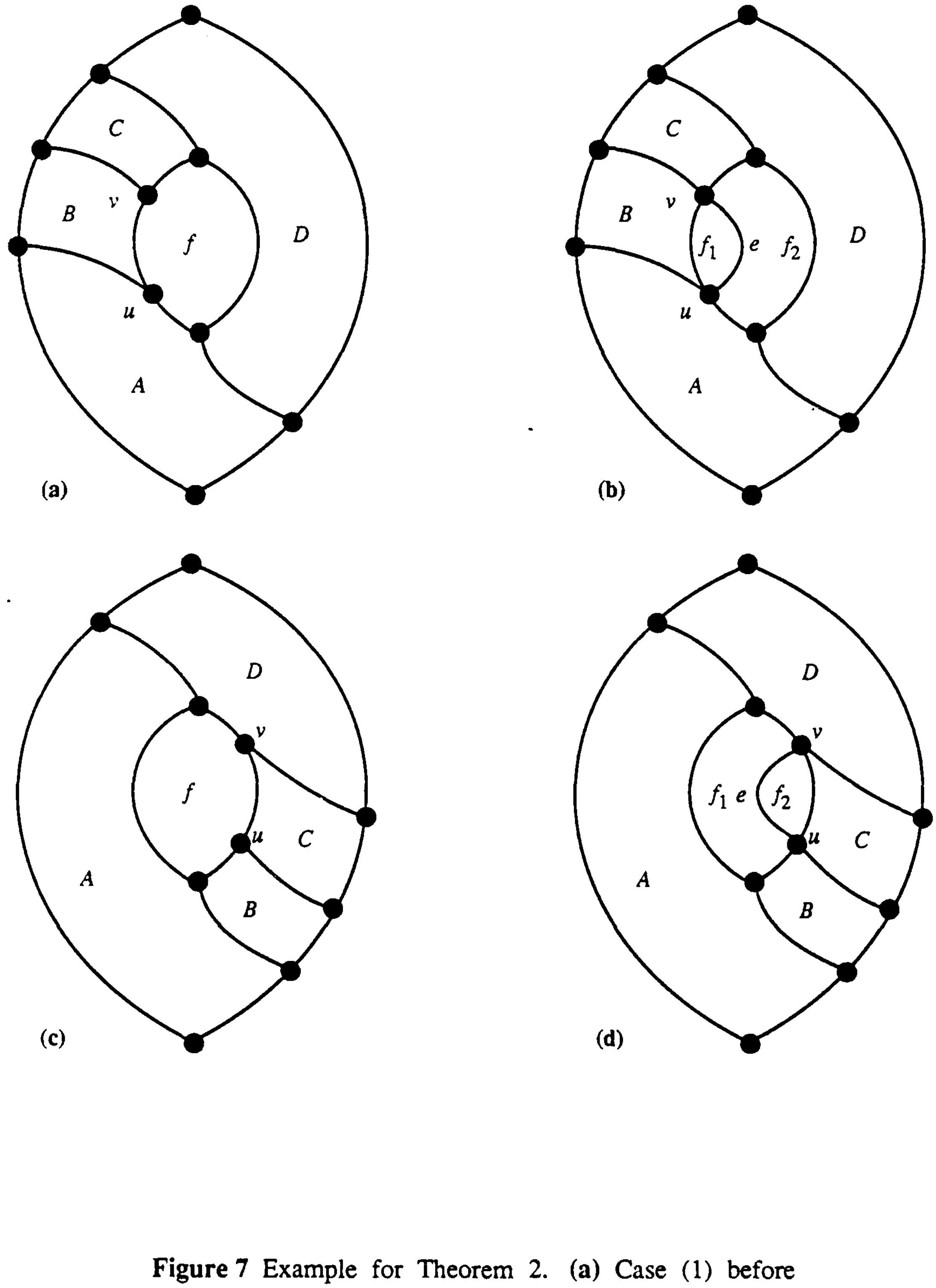

Figure 7 Example for Theorem 2. **(a)** Case (1) before insertion. **(b)** Case (1) after insertion. **(c)** Case (2) before insertion. **(d)** Case (2) after insertion.

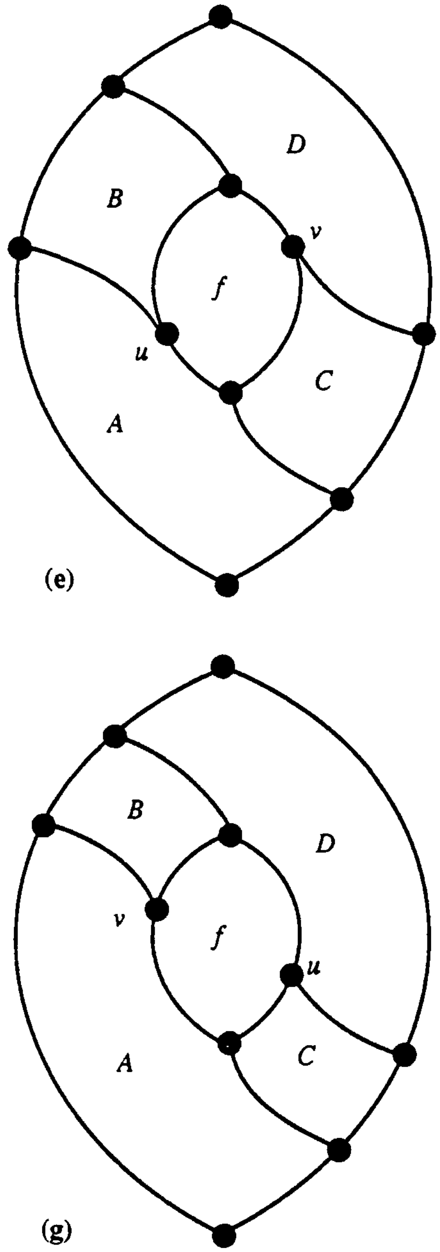

Figure 7 (Continued) Example for Theorem 2. **(e)** Case (3) before insertion. **(f)** Case (3) after insertion. **(g)** Case (4) before insertion. **(h)** Case (4) after insertion.

Theorem 2 shows that the update of the order $<_L$ is a simple syntactic transformation of the left-sequence, consisting of at most four insertions/deletions of elements, and at most one swap of substrings. Since operation *DELETE* is the inverse of operation *INSERT*, the order before and after the deletion can be obtained by reversing the transformations given in Theorem 2. The same situation arises with respect to operations *EXPAND* and *CONTRACT*. We can summarize these results as follows:

Theorem 3 Let G be a planar *st*-graph, and G' be the graph obtained from G after update Π, where Π is one of *INSERT*, *DELETE*, *EXPAND*, or *CONTRACT* operations. Then the left-sequence of G' can be obtained from the left-sequence of G bv means of at most four insertions/deletions of elements, and at most one swap of substrings.

Theorem 3 allows us to design a simple yet efficient data structure for maintaining on-line the orders of a planar *st*-graph G. We represent orders $<_L$ and $<_R$ by means of two balanced binary trees (such as *red-black trees* [28, pp. 52-53]), denoted T_L and T_R. The leaves of T_L and T_R are associated with the elements of G, so that the left-to-right order of the leaves of T_L gives the left-sequence of G, and the left-to-right order of the leaves of T_R gives the right-sequence of G. From Euler's formula, trees T_L and T_R have $O(n)$ nodes, so that their depth is $O(\log n)$.

Definition 5 An *order-query* on a planar *st*-graph G consists of determining, given elements x and y of $V \cup E \cup F$, whether $x <_L y$ or $y <_L x$, and similarly with respect to order $<_R$.

Lemma 10 An order-query can be executed in $O(\log n)$ time.

Proof: The order-query algorithm is as follows. We access the leaves of tree T_L associated with elements x and y, and we trace the paths p_x and p_y from these leaves to the root of T_L. Let node ν be the lowest common ancestor of leaves x and y. We have that $x <_L y$ if and only if the node of p_x immediately preceding ν is the left child of ν. Since paths p_x and p_y have length $O(\log n)$, we obtain the stated time bound. $\square$

Let T be a balanced binary tree. The left-to-right sequence of the leaves of T will be denoted by $\Lambda(T)$. Two basic operations on balanced binary trees are defined as follows:

SPLIT $(T,\lambda;T_1,T_2)$: Construct from tree T two balanced binary trees T_1 and T_2, such that $\Lambda(T_1)$ is the portion of $\Lambda(T)$ from its leftmost leaf to λ, and $\Lambda(T_2)$ is the remaining portion of $\Lambda(T)$. Tree T is destroyed by the operation.

SPLICE $(T_1,T_2;T)$: Construct from the balanced binary trees T_1 and T_2 a new balanced binary tree T such that $\Lambda(T)$ is the concatenation of $\Lambda(T_1)$ and $\Lambda(T_2)$, with $\Lambda(T_1)$ occurring to the left of $\Lambda(T_2)$. Trees T_1 and T_2 are destroyed by the operation.

Let m be the number of leaves of tree T. Standard techniques allow to perform each of the above operations in $O(\log m)$ time [28, pp. 52-53].

As regards the update operations on the planar *st*-graph G, the syntactic transformations on the left- and right-sequence of G correspond to performing $O(1)$ insertions/deletions and *SPLIT/SPLICE* operations on the trees T_L and T_R. Notice that the elements of $V \cup E \cup F$ involved in the update identify the elements of the left-sequence that are inserted, deleted, or are at the boundary of substrings to be swapped. For example, the algorithm for operation *INSERT* is as follows:

Algorithm *INSERT* $(e,u,v,f;f_1,f_2)$

(1) Determine the relative order of u, v, and f in the left-sequence of G by applying the order-query algorithm of Lemma 10. This determines which of the four cases of Theorem 2 applies.

(2) Access leaves u, v, and f in tree T_L and remove them. Also, by means of at most three *SPLIT* operations, construct from T_L four trees associated with substrings A, B, C, and D.

(3) Destroy leaf f and create new leaves f_1 and f_2.

(4) Assemble the updated tree T_L from the leaves u, v, f_1, and f_2, and from the trees associated with A, B, C, and D by a sequence of *SPLICE* operations and insertions. The correct left-to-right order of these constituents is selected according to the specifications of Theorem 2.

(5) Perform the above Steps 1-4 on the right-sequence and tree T_R.

Analogous algorithms can be formulated for the other update operations, and we have:

Theorem 4 The restructuring of trees T_L and T_R after any one of the update operations *INSERT*, *DELETE*, *EXPAND*, and *CONTRACT* can be performed in $O(\log n)$ time.

4. Applications

The general framework for the maintenance of orders $<_L$ and $<_R$ in a planar *st*-graph can be profitably used in three interesting applications: (*i*) dynamic point location in monotone subdivisions, (*ii*) dynamic transitive-closure query in planar *st*-graphs, and (*iii*) dynamic contact-chain query in convex subdivisions.

In this paper we shall consider in detail only Applications (*ii*) and (*iii*). Application (*i*), dynamic planar point location in monotone subdivisions, has been discussed in detail in [22] in a purely geometric setting; here, we simply illustrate how the geometric problem can be reformulated in terms of the planar *st*-graph framework, thus providing a unified viewpoint for these problems.

A monotone subdivision Γ is associated with a planar *st*-graph G such that (see Fig. 8):

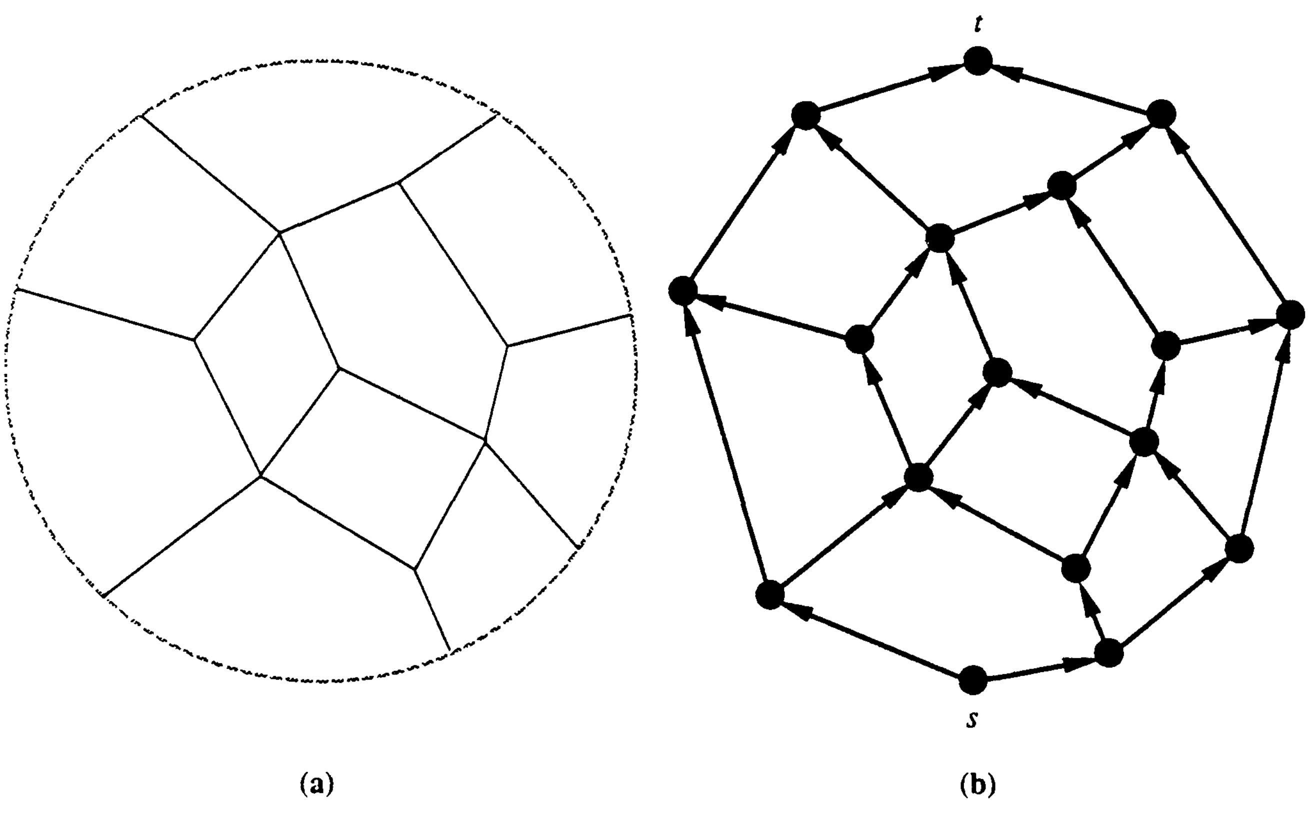

Figure 8 **(a)** Monotone subdivision; **(b)** The planar *st*-graph associated with the monotone subdivision of part (a).

(1) the vertices of G are the vertices of Γ, plus two special vertices s and t, associated with vertices at infinity in the vertical direction;

(2) the arcs of G are associated with the edges of Γ, and oriented from the lower to the upper endpoint; also G contains arcs connecting consecutive vertices of Γ at infinity.

Note that the vertices on the external boundary of G are the vertices of Γ at infinitity, plus s and t.

An order on the regions of Γ (i.e., the faces of G) is obtained as a restriction of, say, $<_L$. This order readily induces a unique set of separating chains of Γ to which the query technique of [17] is immediately applicable, while the results of Section 3 can be used as the theoretical underpinnings for full dynamization of the method (monotonicity-preserving insertions/deletions of edges and vertices). The performance of the resulting dynamic method is expressed by Theorem A of Section 1.

4.1. Transitive-closure query

Recall that a transitive-closure query on a planar st-graph G consists of determining the existence of a directed path between vertices u and v of G. Such query is equivalent to test whether both $u <_L v$ and $u <_R v$ so that, by Lemma 10, it takes $O(\log n)$ time. This establishes Theorem B of Section 1.

A variant of query reports a path between u and v, and can be executed in time $O(\log n + k)$, where k is the number of path edges. First, we query (in $O(\log n)$ time) the existence of a path between u and v. Suppose that such path exists and, say, $u \uparrow v$. We know that the leftmost path from u to t and the leftmost path from v to t have at least one vertex in common. Resorting to a standard DCEL representation of the planar st-graph (see [21, pp. 15-17]), we can trace each of these two paths. Alternating between them one edge at a time, we trace the path between u and t forward from u, and the path between s and v backward from v. In this traversal we mark each visited vertex. The process terminates when we reach a vertex for the second time. If k is the length of the path to be reported, clearly at most $2k$ vertices have been visited by the process. This establishes that the report-type query is executed in time $O(\log n + k)$.

4.2. Contact-chain query

We can reformulate the problem of contact chains by assuming that the reference direction θ is always the x-axis. In this equivalent setting, we have that region r_1 pushes region r_2 if and only if r_1 is to the left of r_2. Hence, the transitive closure of the "push" relation is the same as relation $\rightarrow$, and variations of θ correspond to rotations of the subdivision.

We assume, with negligible loss of generality, that the slopes of the edges are all distinct. (In the case of parallel edges, a virtual perturbation of their slopes achieves this simplifying condition.) Thus, if we continuously rotate the subdivision, only one edge at a time becomes horizontal. An *elementary clockwise rotation* from a given position of Γ is the minimal nonzero clockwise rotation such that an edge becomes horizontal. An *elementary counterclockwise rotation* is correspondingly defined. Thus, a full 2π-rotation of Γ is a sequence of elementary rotations.

Since a convex subdivision Γ is also a monotone subdivision, we consider the planar *st*-graph G associated with Γ, and its dual G^*. It is easy to see that contact chains of Γ are in one-to-one correspondence with paths in the graph G^*.

We consider the following update operations on Γ:

INSERTPOINT $(v,e;e_1,e_2)$: Split the edge $e=(u,w)$ into two edges $e_1=(u,v)$ and $e_2=(v,w)$, by inserting vertex v.

REMOVEPOINT $(v;e)$: Let v be a vertex of degree 2 whose incident edges, $e_1=(u,v)$ and $e_2=(v,w)$, are on the same straight line. Remove v and replace e_1 and e_2 with edge $e=(u,w)$.

INSERTEDGE $(e,u,v,r;r_1,r_2)$: Add edge $e=(u,v)$ inside region r, which is decomposed into regions r_1 and r_2, with r_1 to the left of e and r_2 to the right

REMOVEEDGE $(e,u,v,r_1,r_2;r)$: Remove edge $e=(u,v)$ and merge the regions r_1 and r_2 formerly on the two sides of e into region r. [The operation is allowed only if the subdivision Γ' so obtained is convex.]

ROTATE (δ): Perform an elementary rotation of the subdivision Γ. The binary parameter δ indicates whether the rotation is clockwise or counterclockwise.

To maintain information on the paths of G^*, we use the theoretical framework developed in Sections 2 and 3, and exchange the roles of G and G^*. Operations *INSERTPOINT* and *REMOVEPOINT* on Γ correspond to performing operations *INSERT* and *DELETE* on G^*. Operations *INSERTEDGE* and *REMOVEEDGE* on Γ correspond to performing operations *EXPAND* and *CONTRACT* on G^*. This allows to perform in time $O(\log n)$ contact-chain queries and insertions/deletions of vertices and edges.

With regard to the operation *ROTATE*, let e be the edge of Γ that becomes horizontal at some time during the rotation. The effect of such rotation on G^* is to invert the direction of the dual edge e^* of e (see Fig. 9). Hence, operation *ROTATE* on Γ corresponds to performing a *DELETE* operation on G^*, followed by an *INSERT* operation of the same edge in the reverse orientation.

Let the azimuth of a directed edge be defined counterclockwise with respect to the x-axis, so that it lies in the range $[0,\pi]$. The edge e involved in the rotation can be identified by maintaining a list of the edges of Γ sorted by increasing azimuth. Specifically, the edge involved in a clockwise (counterclockwise) elementary rotation is the first (last) edge of this list, and is moved to the end (front) of the list after the rotation. The list is implemented as a balanced binary tree, so that edges can be efficiently inserted/deleted as specified by the operations *INSERTPOINT*, *REMOVEPOINT*, *INSERTEDGE*, and *REMOVEEDGE*.

In conclusion, all the update operations have $O(\log n)$ time complexity, which establishes Theorem C of Section 1.

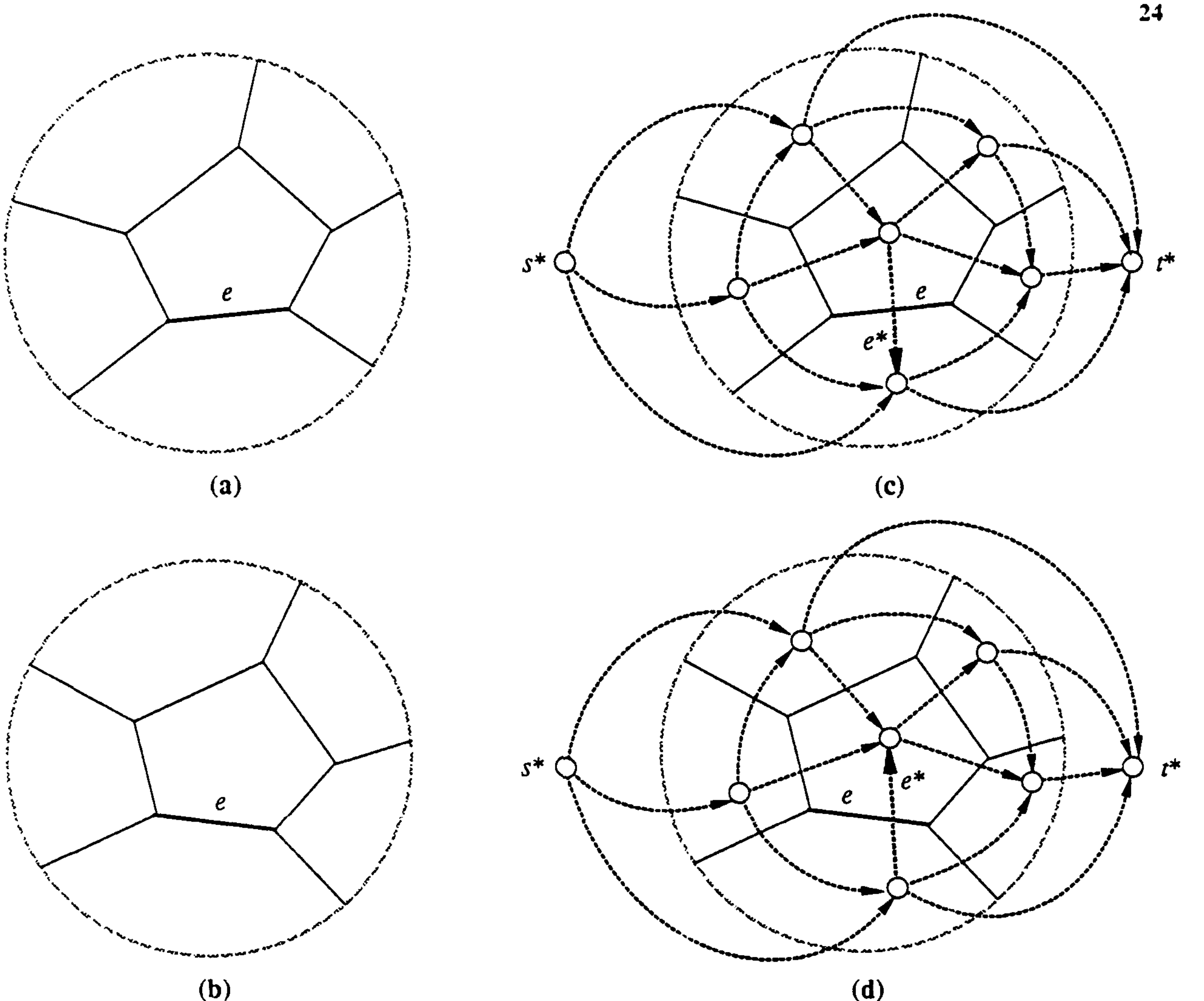

Figure 9 (**a**) Convex subdivision Γ; (**b**) Subdivision Γ after an elementary clockwise rotation (edge e becomes horizontal at some time during the rotation); (**c**) Graph G^* before the rotation; (**d**) Graph G^* after the rotation (the orientation of edge e^* is reversed).

References

[1] J. Bhasker and S. Sahni, "A Linear Algorithm to Find a Rectangular Dual of a Planar Triangulated Graph," *Algorithmica*, vol. 3, no. 2, pp. 247-278, 1988.

[2] G. Birkhoff, "Lattice Theory," *American Mathematical Society Colloquium Publications*, vol. 25, 1979.

[3] B. Chazelle, H. Edelsbrunner, and L.J. Guibas, "The Complexity of Cutting Convex Polytopes," *Proc. 19th ACM Symp. on Theory of Computing*, pp. 66-76, 1987.

[4] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa, "A Linear Algorithm for Embedding Planar Graphs Using PQ-Trees," *J. of Computer and System Sciences*, vol. 30, no. 1, pp. 54-76, 1985.

[5] G. Di Battista and R. Tamassia, "Algorithms for Plane Representations of Acyclic Digraphs," *Theoretical Computer Science*, vol. 58, no. 2,3, 1988 (to appear).

[6] H. Edelsbrunner, L.J. Guibas, and J. Stolfi, "Optimal Point Location in a Monotone Subdivision," *SIAM J. Computing*, vol. 15, no. 2, pp. 317-340, 1986.

[7] S. Even, *Graph Algorithms*, Computer Science Press, 1979.

[8] L.J. Guibas and F.F. Yao, "On Translating a Set of Rectangles," in *Advances in Computing Research, vol. 1*, F.P. Preparata (ed.), JAI Press Inc., pp. 61-77, 1983.

[9] M.Y. Hsueh and D.O. Pederson, "Computer-Aided Layout of LSI Circuit Building-Blocks," *Proc. IEEE Int. Symp. on Circuits and Systems*, pp. 474-477, 1979.

[10] G.F. Italiano, "Amortized Efficiency of a Path Retrieval Data Structure," *Theoretical Computer Science*, vol. 48, pp. 273-281, 1986.

[11] G.F. Italiano, "Finding Paths and Deleting Edges in Directed Acyclic Graphs." *Information Processing Letters*, 1988 (to appear).

[12] R. Jayakumar, K. Thulasiraman, and M.N.S Swamy, "An Optimal Algorithm for Maximal Planarization of Nonplanar Graphs," *Proc. 1986 IEEE Int. Symp. on Circuits and Systems*, pp. 1237-1240, 1986.

[13] R. Jayakumar, K. Thulasiraman, and M.N.S Swamy, "Planar Embedding: Linear-Time Algorithms for Vertex Placement and Edge Ordering," *IEEE Trans. on Circuits and Systems*, vol. 35, no. 3, pp. 334-344, 1988.

[14] T. Kameda, "On the Vector Representation of the Reachability in Planar Directed Graphs," *Information Processing Letters*, vol. 3, no. 3, pp. 75-77, 1975.

[15] D. Kelly and I. Rival, "Planar Lattices," *Canadian J. Mathematics*, vol. 27, no. 3, pp. 636-665, 1975.

[16] D. Kelly, "Fundamentals of Planar Ordered Sets," *Discrete Mathematics*, vol. 63, pp. 197-216, 1987.

[17] D.T. Lee and F.P. Preparata, "Location of a Point in a Planar Subdivision and its Applications," *SIAM J. Computing*, vol. 6, no. 3, pp. 594-606, 1977.

[18] A. Lempel, S. Even, and I. Cederbaum, "An Algorithm for Planarity Testing of Graphs," *Theory of Graphs, Int. Symposium*, Rome, pp. 215-232, 1966.

[19] R.M. Otten and J.G. van Wijk, "Graph Representations in Interactive Layout Design," *Proc. IEEE Int. Symposium on Circuits and Systems*, New York, pp. 914-918, 1978.

[20] T. Ozawa and H. Takahashi, "A Graph-planarization Algorithm and its Applications to Random Graphs," *Lecture Notes in Computer Science* (*Graph Theory and Algorithms*), vol. 108, Springer-Verlag, pp. 95-107, 1981.

[21] F.P. Preparata and M.I. Shamos, *Computational Geometry*, Springer-Verlag, 1985.

[22] F.P. Preparata and R. Tamassia, "A Fully Dynamic Planar Point Location Technique," Technical Report ACT-84, Coordinated Science Laboratory, Univ. of Illinois at Urbana-Champaign, 1987.

[23] I. Rival and J. Urrutia, "Representing Orders on the Plane by Translating Convex Figures," Technical Report TR-87-05, Univ. of Ottawa, 1987.

[24] P. Rosenstiehl and R.E. Tarjan, "Rectilinear Planar Layouts of Planar Graphs and Bipolar Orientations," *Discrete & Computational Geometry*, vol. 1, no. 4, pp. 342-351, 1986.

[25] R. Tamassia and I.G. Tollis, "A Unified Approach to Visibility Representations of Planar Graphs," *Discrete & Computational Geometry*, vol. 1, no. 4, pp. 321-341, 1986.

[26] R. Tamassia and I.G. Tollis, "Efficient Embedding of Planar Graphs in Linear Time," *Proc. IEEE Int. Symp. on Circuits and Systems*, Philadelphia, pp. 495-498, 1987.

[27] R. Tamassia, "A Dynamic Data Structure for Planar Graph Embedding," *Lecture Notes in Computer Science* (*Proc. 15th International Colloquium on Automata, Languages and Programming, Tampere, Finland, 1988*), Springer-Verlag, 1988 (to appear).

[28] R.E. Tarjan, "Data Structures and Network Algorithms," *CBMS-NSF Regional Conference Series in Applied Mathematics*, vol. 44, Society for Industrial Applied Mathematics, 1983.

[29] S. Wimer, I. Koren, and I. Cederbaum, "Floorplans, Planar Graphs, and Layouts," *IEEE Trans. on Circuits and Systems*, vol. 35, no. 3, pp. 267-278, 1988.

[30] S. Wismath, "Characterizing bar line-of-sight graphs," *Proc. ACM Symposium on Computational Geometry*, Baltimore, Maryland, pp. 147-152, 1985.

[31] D. Woods, "Drawing Planar Graphs," Ph.D. Thesis (Technical Report STAN-CS-82-943), Computer Science Dept., Stanford Univ., 1982.

END
DATE
9-88
DTIC